An Exposure Model for Supersingular Isogeny Diffie-Hellman Key Exchange

Brian Koziel[1], Reza Azarderakhsh[2], and David Jao[3,4]

[1]Texas Instruments Inc., Dallas, Texas, USA
kozielbrian@gmail.com
[2]Computer and Electrical Engineering and Computer Science Department and I-SENSE, Florida Atlantic University
razarderakhsh@fau.edu
[3]Centre for Applied Cryptographic Research, University of Waterloo
djao@uwaterloo.ca
[4]evolutionQ Inc., Waterloo, Ontario, Canada
david.jao@evolutionq.com

Abstract. In this work, we present an exposure model for the isogeny computation in the quantum-resistant supersingular isogeny Diffie-Hellman (SIDH) key exchange protocol. Notably, we propose this exposure model to characterize the severity of new attacks that force an SIDH user to divulge certain intermediate values. In our model, we show how an attacker can break SIDH by discovering an intermediate kernel point and its corresponding curve. To strengthen an SIDH-user against the exposure of intermediate values, we propose a random curve isomorphism that is performed just before the large-degree isogeny. We show that this countermeasure is computationally inexpensive compared to the whole of SIDH and can still operate with the Kirkwood et al. validation model that allows a static-key user to ensure the first round of the other party was performed honestly. The goal of this paper is to present an additional protection against future attacks for implementations of SIDH.
Key Words: Post-quantum cryptography, isogeny-based cryptography, exposure model

1 Introduction

The threat of the emergence of a quantum computer has put the security and privacy of society's digital data in jeopardy. In a recent announcment at PQCrypto 2016, NIST announced a preliminary standardization plan to standardize post-quantum cryptographic (PQC) algorithms that are resistant to an attacker armed with both classical and quantum computers [4]. Lattice-based cryptosystems, code-based cryptosystems, multivariate cryptosystems, and the McEliece cryptosystem are known to be among the top contenders for PQC standardization.

Recently, isogeny-based cryptography has emerged as a dark horse candidate since the supersingular isogeny Diffie-Hellman (SIDH) key exchange protocol features the smallest known PQC keys and also features forward secrecy. Proposed

by Jao and De Feo [14], this cryptosystem utilizes isogenies between supersingular elliptic curves to move between supersingular elliptic curve isomorphism classes in such a way that two parties arrive at curves with the same j-invariant. This key exchange protocol is currently difficult even for quantum computers to crack, as the best known quantum attack to compute isogenies between supersingular elliptic curves has complexity $O(\sqrt[6]{p})$ for a field of characteristic p [14].

However, since isogeny-based cryptography has only been popularized in the previous several years, various applications, implementations, and in-depth security analyses from the cryptographic research community are still underway. Namely, several recent advances include key compression [2,5], digital signatures [11,26], static-static key agreement [3], and efficient software and hardware implementations [7,1,6,20,18,19,13]. A few attacks on isogeny-based cryptography have been proposed, notably protocol attacks [10], side-channel attacks [17], and fault attacks [24,12].

Here, we provide a so-called "exposure" model for the isogeny computation critical to SIDH. For this model we consider the impact on the security assumptions of SIDH when certain intermediate values have been exposed or leaked in some way. Since isogeny-based cryptography and SIDH are still in their infancy compared to other quantum-resistant schemes we provide this model as a way to account for attacks that are discovered in the future.

Our contributions:

- We propose for the first time an exposure model in the large-degree isogeny computations of SIDH.
- We introduce a random pre-isogeny isomorphism as an additional side-channel countermeasure for the large-degree isogeny computation.
- We show that this isomorphism is inexpensive and can still operate seamlessly with the Kirkwood et al. [16] validation model.

2 Preliminaries

Here, we briefly reiterate key components of isogeny-based cryptography that act as a foundation for SIDH. For a complete background in elliptic curve theory, we point the reader to [22].

2.1 Elliptic Curve Theory

Elliptic Curves: An elliptic curve defined over a finite field, $\mathbb{F}_q$, can be written in its short Weierstrass form as:

$$E/\mathbb{F}_q : y^2 = x^3 + ax + b$$

where a,$b \in \mathbb{F}_q$. An elliptic curve is composed of all points (x, y) that satisfy the above equation as well as the point at infinity. This forms an abelian group over

point addition, the underlying basis of the scalar point multiplication in elliptic curve cryptography.

In addition to short Weierstrass form, other curve forms have been researched, such as Edwards [8] and Montgomery [21] curves. Depending on the application, these curves can provide various efficiency and security benefits. They still fit for elliptic curve cryptography applications because there has been shown to be an equivalence between these curve forms and the short Weierstrass form [8,21]. More specifically, every Montgomery and Edwards curve has an equivalent short Weierstrass curve, but a short Weierstrass curve may not have an equivalent Montgomery or Edwards curve.

Isogenies: We define an isogeny over a finite field, $\mathbb{F}_q$, $\phi: E \to E'$ as a non-constant rational map over $\mathbb{F}_q$, where ϕ is a group homomorphism from $E(\overline{\mathbb{F}_q})$ to $E'(\overline{\mathbb{F}_q})$. Isogenies are essentially a way to jump from one elliptic curve isomorphism class to another. Specifically, we are looking at supersingular elliptic curves, which have an endomorphism ring with $\mathbb{Z}$-rank equal to 4. Supersingular curves can be defined over $\mathbb{F}_{p^2}$, for a given prime p. For every prime $\ell \neq p$, there exist $\ell + 1$ unique isogenies up to isomorphism of degree ℓ originating from a given supersingular curve. We can compute these unique isogenies over a kernel, κ, such that $\phi: E \to E/\langle\kappa\rangle$ by using Vélu's formulas [25].

The j-invariant of an elliptic curve defines various complex properties of the elliptic curve and also acts as an identifier for its corresponding elliptic curve isomorphism class. Over the short Weierstrass form, we can compute the j-invariant as follows:

$$j(E) = 1728\frac{4a^3}{4a^3 + 27b^2}$$

Elliptic curves that share the same j-invariant are isomorphic. Thus, elliptic curves in the same isomorphism class share various complex properties. There are a total of q isomorphism classes over $\mathbb{F}_q$, or an isomorphism class for each element in $\mathbb{F}_q$.

An elliptic curve isogeny performs a mapping from an elliptic curve E to another elliptic curve E'. Isogenies of degree one are isomorphisms, where E and E' will stay in the same isomorphism class and thus share the same j-invariant. Isogenies with a degree higher than one will move across isomorphism classes, where E and E' will no longer share the same j-invariant. In addition, an isogeny can also be applied to either an elliptic curve or specified points on an elliptic curve. Computing an isogeny is where we find the mapping from one elliptic curve to another and correspondingly update the elliptic curve coefficients. Evaluating an isogeny is where we push a point from one elliptic curve to another based on the mapping from an isogeny computation.

2.2 Large-Degree Isogeny Computation

We can break large-degree isogenies of the form ℓ^e by performing multiple isogenies of degree ℓ. These are performed iteratively. Consider computing an isogeny

of degree ℓ^e on the supersingular elliptic curve E with a point R, of order ℓ^e, as the kernel point. We efficiently compute $\phi: E \to E/\langle R \rangle$ by decomposing ϕ into a chain of degree ℓ isogenies, $\phi = \phi_{e-1} \circ \cdots \circ \phi_0$. We initialize $E_0 = E$ and $R_0 = R$, and perform each isogeny as follows:

$$E_{i+1} = E_i/\langle \ell^{e-i-1} R_i \rangle, \ \phi_i : E_i \to E_{i+1}, \ R_{i+1} = \phi_i(R_i)$$

As is shown in Figure 1, the large-degree isogeny computation can be visualized as traversing an acyclic graph in the shape of a triangle with each node representing various important multiples and isogenies of the kernel point. Each node represents an intermediate kernel point. The large-degree isogeny computation starts at the top of the graph with the secret kernel point R_0. Performing a point multiplication by ℓ moves to left and evaluating an isogeny of degree ℓ with the point moves to the right. The large-degree isogeny can be efficiently computed by computing an isogeny of degree ℓ at each of the green nodes at the bottom level, or rather, at each of the torsion points $[\ell^{e-i-1}]R_i$ for $i < e$. Thus, an optimal strategy to compute the large-degree isogeny will perform the most efficient traversal to the bottom of the graph.

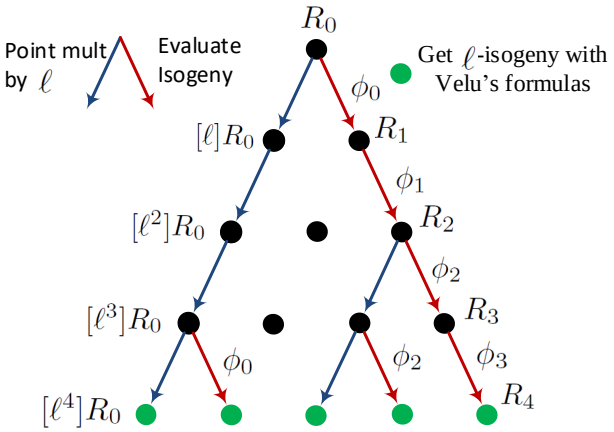

Fig. 1. Acylic graph depicting a large-degree isogeny computation with an optimal strategy.

Introduced in [7], the traversal of this acyclic graph to its leaves can be broken down into a combinatorial problem. A strategy represents a series of computations to traverse the directed acyclic graph to its leaves. By identifying the costs to move left and right on the triangle, one can measure the total cost of various strategies to find the optimal strategy. Interestingly, [7] shows that an optimal strategy is composed of two optimal sub-strategies. Thus, one can form an optimal strategy with the least cost of traversal by combining optimal sub-strategies in a divide and conquer fashion. This method does require the storage

of intermediate points to act as pivots, but the reduction of time complexity from $O(e^2)$ to $O(e\log e)$ is a great boon to performance. Both [7] and [6] feature source code that solve this dynamic programming problem.

2.3 Supersingular Isogeny Diffie-Hellman

Public Parameters: The SIDH protocol is a public-key cryptosystem where Alice and Bob want to agree on a shared key over a public channel that can be monitored by third-parties. To initiate the protocol, several public parameters must be determined. Alice and Bob first agree on a prime p of the form $\ell_A^{e_A}\ell_B^{e_B}f \pm 1$, where ℓ_A and ℓ_B are small primes, e_A and e_B are positive integers, and f is a small cofactor to make the number prime. Over the finite field generated by this prime, a supersingular elliptic curve $E_0(\mathbb{F}_{p^2})$ is selected and two torsion bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ are found that generate $E_0[\ell_A^{e_A}]$ and $E_0[\ell_B^{e_B}]$, respectively. Lastly, Alice chooses two private keys $m_A, n_A \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ such that both are not divisible by ℓ_A and Bob likewise chooses two private keys $m_B, n_B \in \mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$ such that both are not divisible by ℓ_B.

Protocol: The SIDH protocol is composed of two rounds consisting of a double-point multiplication to generate a secret kernel $R = mP + nQ$ and a large-degree isogeny computation over that kernel $\phi : E \to E/\langle R\rangle$. In the first round of SIDH, Alice and Bob each compute their secret kernel, $R_A = \langle [m_A]P_A + [n_A]Q_A\rangle$ and $R_B = \langle [m_B]P_B + [n_B]Q_B\rangle$, respectively. Alice and Bob perform a large-degree isogeny to move to a new supersingular elliptic curve class, $\phi_A : E_0 \to E_A = E_0/\langle R_A\rangle$ and $\phi_B : E_0 \to E_B = E_0/\langle R_B\rangle$, respectively. As they perform this isogeny, they also compute the image of the opposite party's basis points under the new curve, $\{\phi_A(P_B), \phi_A(Q_B)\} \subset E_A$ of the basis $\{P_B, Q_B\}$ for $E_0[\ell_B^{e_B}]$ on Alice's side and $\{\phi_B(P_A), \phi_B(Q_A)\} \subset E_B$ of the basis $\{P_A, Q_A\}$ for $E_0[\ell_A^{e_A}]$ on Bob's side. At the end of the first round, the values $E_A, E_B, \phi_A(P_B), \phi_A(Q_B)$, $\phi_B(P_A)$, and $\phi_B(Q_A)$ are exchanged over a public channel. The second round proceeds similarly, but over the new torsion basis points received from the opposite party. Alice and Bob compute a second double-point multiplication, $R_{AB} = \langle [m_A]\phi_B(P_A) + [n_A]\phi_B(Q_A)\rangle$ and $R_{BA} = \langle [m_B]\phi_A(P_B) + [n_B]\phi_A(Q_B)\rangle$, respectively. Alice and Bob then compute a second isogeny, $\phi'_A : E_B \to E_{AB} = E_B/\langle R_{AB}\rangle$ and $\phi'_B : E_A \to E_{BA} = E_A/\langle R_{BA}\rangle$, respectively. Since curves E_{AB} and E_{BA} are isomorphic, Alice and Bob can use the j-invariant as the shared secret [14].

Security: The security of the SIDH protocol relies on the difficulty to compute isogenies between supersingular elliptic curves. Let us consider a graph of all supersingular curves under $\mathbb{F}_{p^2}$, where each vertex represents an isomorphism class and the edges represent an isogeny of degree ℓ. To the casual observer, the SIDH protocol performs a large walk composed of many perceptibly random steps in the graphs of degree ℓ_A and ℓ_B to arrive at a specific isomorphism class. Thus, the SIDH protocol is protected by the infeasibility to discover a path between two specified vertices. For instance, if an attacker can discover the path $\phi_A : E_0 \to E_A$, then he can surely use the same path to perform the computation $\phi'_A : E_B \to E_{AB}$ to find Alice and Bob's shared key. As noted in [7], the best

classical and quantum attack is based on the claw finding problem. Galbraith and Stolbunov [9] describes the most efficient attack for classical computers. To break an isogeny of degree $\ell_A^{e_A}$ between E_0 and E_A with a classical computer, an attacker can construct two trees consisting of all curves isogenous to E_0 and E_A, respectively, from isogenies of degree $\ell_A^{e_A/2}$. The attacker then looks for a curve lying in both trees, as a meet-in-the-middle approach . This attack has time complexity $O(\ell_A^{e_A/2})$ or $O(\sqrt[4]{p})$ for the SIDH protocol [7]. Tani [23] notes an even faster attack for quantum computers, relying on quantum walks, with time complexity $O(\ell_A^{e_A/3})$ or $O(\sqrt[6]{p})$ for the SIDH protocol.

3 Proposed Isogeny Computations Exposure Model

In this section, we propose an exposure model of the large-degree isogeny computation. We define this exposure model as a method by which we analyze the security of a cryptosystem if any intermediate computations are exposed to an outside party. Specifically, we examine how much of the SIDH cryptosystem is broken if certain pieces of information from the isogeny computation are divulged. Since the large-degree isogeny computation is relatively new and critical to the SIDH protocol, the model is left in a general manner to account for any new attacks.

Model construction. The latest implementations of the SIDH protocol have primarily utilized the Jao, De Feo, and Plût model [7] to compute the large-degree isogeny computation ℓ^e with $O(e \log e)$ point multiplications by ℓ and isogeny evaluations of degree ℓ. Throughout the computations, isogeny mappings and point multiples of the hidden kernel point, R, are utilized to compute the torsion points $[\ell^{e-i-1}]R_i = \phi_{i-1} \circ \phi_{i-2} \circ \cdots \circ \phi_0([\ell^{e-i-1}]R)$. To generalize this representation for intermediate points in the isogeny computation, we denote the variable j to be the number of point multiplications by ℓ and variable k to be the number of isogenies that the point or curve has been pushed through. Thus, a general intermediate kernel point is represented as $\phi_{k-1} \circ \phi_{k-2} \circ \cdots \circ \phi_0([\ell^j]R)$. To make this compact, we represent the isogeny evaluation notation as $\phi_{k-1:0}$. In addition to the hidden kernel point, we also represent the intermediate curve, E_k, as $\phi_{k-1:0} : E_0 \to E_k$. In this case, E_e represents the resulting curve from the large-degree isogeny computation. In the first round, Alice and Bob will also push the other party's torsion basis through each isogeny, or $\phi_{k-1:0}(P)$ and $\phi_{k-1:0}(Q)$ in the general case. In the following sections, we primarily focus on the exposed values in the first round. It can be assumed that if the isogeny decisions used in the second round are divulged that they will be identical in the first round. Further, any exposed values from the first round can be used to retrieve the shared key from the second round.

3.1 Exposure Classes

Intermediate curve. First, we consider the exposure of some intermediate curve E_k, which directly impacts the security assumption. Consider that Alice's implementation has unknowingly exposed E_k. An attacker no longer has

Algorithm 1 Proposed method to retrieve SIDH private keys with some exposed values

Input: SIDH protocol over base curve E_0
Party's torsion basis: P_A,Q_A over isogenies of degree ℓ_A
Exposure of intemediate kernel point $S = \phi_{k-1:0}([\ell_A^j m]P_A + [\ell_A^j n]Q_A)$ on curve E_k
1. Compute isogeny $\phi_{k-1:0} : E \to E_k$ for which S lies on
2. Apply isogeny to torsion basis $\phi_{k-1:0}(P_A)$,$\phi_{k-1:0}(Q_A)$
3. Determine order of S, which is $\ell_A^{e_A - j}$
4. Perform generalized elliptic curve discrete log:
$\phi_{k-1:0}([\ell_A^j m]P_A + [\ell_A^j n]Q_A) = \phi_{k-1:0}([m']P_A) + \phi_{k-1:0}([n']Q_A)$
5. Use isogeny brute-force information and exhaustive search of size ℓ^j
to retrieve m and n from m' and n'
6. **return** secret keys m, n

to compute the large isogeny $\phi_A : E_0 \to E_A$. Instead, he can break it into two, smaller isogeny computations $\phi_{k-1:0} : E_0 \to E_k$ and $\phi_{e_A-1:k} : E_k \to E_A$. Thus, the difficulty of this assumption becomes the difficulty of the larger isogeny to compute, or $\sqrt[3]{\text{MAX}(\ell^k, \ell^{e_A-k})}$ in the quantum case. The absolute worst case is if $E_{e_A/2}$ is discovered, upon which the security assumption is cut in half.

Interestingly, this exposure class has already been attacked through the use of the loop-abort attack proposed by Gélin and Wesolowski [12]. In this particular case, the large-degree isogeny computation is generally done iteratively. Thus, by forcing a fault on the loop counter, an implementation may divulge the intermediate curve E_k. As Gélin and Wesolowski propose, this loop-abort attack can be performed iteratively to reveal each isogeny decision and thus the full isogeny. In our description of this exposure class, we generalize the exposure of these intermediate curves to how much easier the security assumption becomes.

Intermediate kernel point. Second, we consider the exposure of the kernel point at some intermediate stage, $\phi_{k-1:0}([\ell^j]R)$. This can completely break the SIDH security assumption, as the kernel point is intended to stay secret and can be used to directly compute the isogeny. However, the intermediate kernel point must be associated with its curve E_k. An attacker can retrieve that hidden curve by brute-forcing all possible isogenous curves (based on the system's choice of Vélu's formulas [25]) out to some defined bound i. If the attacker finds some isogenous curve with the intermediate kernel point on it, then the attacker has already computed several of the isogenies and can use the specific node on the isogeny computation graph (Figure 2a) to compute the remaining isogenies. Thus, an attacker can identify the unknown path from E_0 to E_A with the combination of an intermediate kernel point and its corresponding curve. The attack from this point of view is a brute-force attack to discover $\phi_{k-1:0}$ with complexity $O(\ell^k)$.

However, we remark that this can lead to an even worse attack:

Remark 1. The exposure of an intermediate kernel point and its supersingular elliptic curve can be used to recover the party's private keys.

Recall that the generalized discrete logarithm is simple for SIDH, even without the use of quantum computers, as it is already utilized in key compression [2,5]. One can calculate the order of the kernel point that is exposed, this indicates how many point multiplications and isogeny evaluations by ℓ have been performed. To setup a generalized discrete logarithm for the secret keys, an attacker computes the isogeny $\phi_{k-1:0} : E_0 \to E_k$ and pushes the party's basis points through the isogeny to retrieve $\phi_{k-1:0}(P)$ and $\phi_{k-1:0}(Q)$. With a known torsion basis the generalized discrete logarithm will return scalars that are directly associated with the initial private keys, notably $m' = m\ell_A^j$ and $n' = n\ell_A^j$. Assuming that the generalized discrete log returns values modulo the order of the group, the attacker now has a large portion of the key, or rather the key modulo ℓ^{e-k-j}. For a key of k isogeny decisions, these scalars represent decisions $k-j$ downto 0. The full key is m_A added to some multiple of the order of the group, which can be found through exhaustive search to find the missing key bits for ℓ^j along with the brute-forced isogeny decisions for the last ℓ^k bits. We demonstrate this attack step-by-step in Algorithm 1. The most difficult step is either computing the supersingular isogeny $\phi_{k-1:0}$, with difficulty $O(\ell^k)$, or performing exhaustive search on the point multiples j, with difficulty $O(\ell^j)$. Even if j is very high, solving the first isogenies will already weaken the security of the cryptosystem as essential isogeny decisions have been found.

One optimization to this brute-force attack could be forming an equation with the leaked point for the coefficients a and b of the short Weierstrass curve. One point would not solve the equation for the coefficients, but such an equation creates a constraint between a and b that can then be used in the j-invariant formula to "cross-out" some j-invariants that do not fit. However, if two intermediate kernel points on the same curve are exposed (i.e., pivot points to efficiently perform the isogeny computation illustrated in Figure 1), then an attacker can easily solve the elliptic curve equation for a and b and find the corresponding curve that way. This is slightly different than the attack proposed in Algorithm 1 as the path from the initial node to the intermediate node has not been determined. However, with an intermediate kernel point and corresponding curve, an attacker can once again perform the remaining isogenies to get the latter portion of the isogeny walk.

Intermediate basis point. Lastly, we consider the impact of exposing intermediate basis points of the opposite party, $\phi_{k-1:0}(P)$. Unlike exposing an intermediate kernel point, the starting basis points are known. Thus, the exposure of $\phi_{k-1:0}(P)$ can be used in conjunction with P and E_0 to determine which supersingular isogenies were performed as a result of the hidden kernel point. However, this also turns into a brute-force solution, as an attacker tests possible curves E_k and determines if the point $\phi_{k-1:0}(P)$ exists on it. As an example, if $k = 1$, then there are $\ell + 1$ possible isogenies from the starting curve and an attacker can easily brute-force them. This reduces the security of the protocol

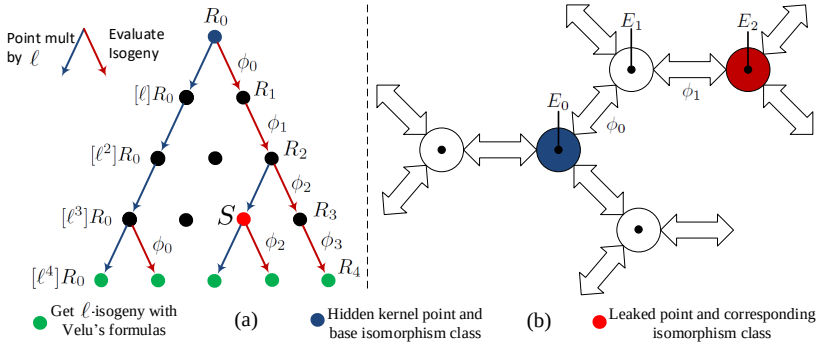

Fig. 2. Visualization of a exposure scenario when $S = \phi_{1:0}([2]R_A)$ has been exposed in the computation of an isogeny of degree 2^5. (a) Large-degree isogeny computation after an intermediate kernel point has been exposed (b) Sub-graph representing the space of all isogenies of degree 2 under a given field, $\mathbb{F}_{p^2}$. The vertices (circles) represent an isomorphism class, of which all curves within the class share the same j-invariant.

by a factor of ℓ. Thus, for a generic k, the security of the protocol is reduced by a factor ℓ^k if the brute-force can be efficiently performed.

Similar to the discussion of using the kernel point in the elliptic curve equation, the exposure of an intermediate basis point can also be used to remove some j-invariant combinations and divulging both basis points can be used to recover the corresponding curve. In fact, any two divulged points could be used to recover the intermediate curve, but divulging the kernel point is even more of a disaster as further isogenies could be computed.

Incomplete pieces of information. There are more cases, such as the exposure of the x-coordinate of an intermediate kernel point. From a security standpoint, it is difficult to quantify the loss, if any, of security in the underlying assumption. However, an attacker can compile a list of various pieces of information to expose portions of the computation.

3.2 Exposure Model Scenario

Let us consider a simple scenario to illustrate the attack given in Algorithm 1. For her side of the protocol, Alice performs an isogeny of degree 2^5 using a hidden kernel point that she previously computed from the double-point multiplication $R_A = \langle [m_A]P_A + [n_A]Q_A \rangle$, where $\{P_A, Q_A\}$ is her torsion basis and $\{m_A, n_A\}$ are secret keys that she generated through her random number generator. Oscar, a malicious third-party, discovers an attack on Alice's implementation and discovers the intermediate kernel point $S = \phi_{1:0}([2]R_A)$. The left-side of Figure 2 visualizes this attack scenario in terms of the large-degree isogeny computation and the right-side visualizes a graph where the vertices represent isomorphism classes and the edges represent isogenies.

At this point, Oscar is unsure of which node on the large-degree isogeny computation he has compromised. However, he knows that Alice and Bob are using a standard library for the SIDH implementation and thus also knows which set of Vélu's formulas are used. Oscar's goal is to find the curve E_k for which S lies on. We refer to Algorithm 1 to perform the attack. For Step 1, Oscar proceeds in a breadth-first search from the base curve, E_0, to check if the nearby isogenous curves contain S. Since Vélu's formulas are deterministic, Oscar can determine exactly which curve within the isomorphism class that Alice would have moved to with an isogeny. If this step is feasible (i.e., k is relatively small), then Oscar now has both an intermediate kernel point and its corresponding elliptic curve. Step 2, Oscar pushes Alice's basis points through the isogenous mapping to retrieve $\phi_{k-1:0}(P_A)$,$\phi_{k-1:0}(Q_A)$. Step 3, Oscar finds the order of S, which is $\ell_A^{e_a-j}$. Here, j represents the number of point multiplications by $\ell = 2$ that Alice applied to the original kernel point. In this case, $j = 1$. Step 4, Oscar solves the generalized discrete logarithm, $S = \phi_{k-1:0}([m_A]P_A)+\phi_{k-1:0}([n_A]Q_A)$. Lastly, in Step 5, Oscar retrieves Alice's private keys m, n by combining the brute-forced isogeny decisions and performing an exhaustive search to find the most significant ℓ^j bits of Alice's private keys m, n.

3.3 Exposure Model For Montgomery Coordinates

In the above, we assumed a generic model for large-degree isogenies. In the current implementations of SIDH [7,6,18,20], Montgomery coordinates with arithmetic on the Kummer line [21] have been popular because they provide fast point multiplication and isogeny calculations. In this arithmetic, a point is represented only by its x-coordinate on the Kummer line, which provides for an efficient point multiplication ladder [21], isogeny arithmetic over base degrees 2 and 3 [7], and also "projectivized" isogeny arithmetic over base degrees 3 and 4 [6]. A Montgomery curve defined over $\mathbb{F}_q$ is written in the form $E/\mathbb{F}_q : by^2 = x^3 + ax^2 + x$. The "projectivized" version of the Montgomery curve incorporates a C coefficient that acts as a denominator to avoid inversions: $E/\mathbb{F}_q : By^2 = Cx^3 + Ax^2 + Cx$, where $a = A/C$ and $b = B/C$. In the projectivized curve arithmetic, the B coefficient is not needed and discarded.

When applying the exposure model to Montgomery curve arithmetic, the Kummer representation and projectivized isogeny arithmetic make it more difficult to identify the divulged values. In the original exposure model, the leakage of a point indicated exposing both the x and y-coordinates that could be used to identify an isogenous curve. With the Kummer representation, simply discovering an x-coordinate does not reveal enough useful information as the x-coordinate lies on the target curve and a large number of twists. Indeed, the attack in Algorithm 1 now requires both the exposure of an intermediate kernel point and its identifying a coefficient so that an attacker can identify the isomorphism class of the exposed point and compute the isogeny between the extracted isomorphism class and the initial isomorphism class.

Although the use of projectivized Montgomery isogeny arithmetic helps mitigate the exposure problem, it does not prevent the isogeny exposure attack as

the isogeny arithmetic is deterministic and can be reversed to break the cryptosystem's security assumption.

4 Exposure Model Countermeasure

Here, we discuss the applications of a random curve isomorphism as an additional defense against this exposure scenario.

4.1 Random Pre-Isogeny Curve Isomorphism

In an effort to obfuscate any points throughout the large-degree isogeny computation, we propose utilizing a random curve isomorphism at the beginning of a large-isogeny computation in the SIDH protocol. Unfortunately, we cannot obfuscate an exposed curve as the j-invariant can be used to identify its vertex in the graph of all isomorphism classs. SIDH utilizes Vélu's formulas [25] to compute isogenies between elliptic curves. Given a specific elliptic curve, the choice of Vélu's formula will produce the same isogenous elliptic curve in the new isomorphism class since the formulas are deterministic. However, by scaling the elliptic curve, such as by an isomorphism, the resulting isogenous elliptic curve is different, but still within the isomorphism class. As is noted in [22] (III.1.4(b) and III.1.7(c)), the size of a single isomorphism class of a curve $E/\mathbb{F}_q$ is approximately $q/6$, so for supersingular curves defined over $\mathbb{F}_{p^2}$, there are approximately $p^2/6$ unique isomorphisms. Thus, by performing a random isomorphism at the beginning of a round, we are obfuscating the relationship between R and $\phi_{k-1:0}([\ell^j]R)$ as well as the relationship between P and $\phi_{k-1:0}(P)$.

With this new randomization, an attacker cannot identify E_0, P, or Q. Thus, the brute-force attack that determines E_k from the knowledge of E_0, P, and $\phi_{k-1:0}(P)$ becomes exponentially harder. An attacker must now brute-force both the random curve isomorphism and the isogeny between E_0 and E_k. One can go further and compute an isomorphism after every isogeny evaluation, but this offers little additional resistance, becomes increasingly expensive, and the curve can no longer take advantage of some efficient point arithmetic in the isogeny computation.

4.2 Complexity Analysis of Isomorphism Countermeasure

The random pre-isogeny isomorphism primarily serves as a way to protect any intermediate values that are exposed or divulged throughout the SIDH key exchange protocol. The cost of a random isomorphism is the cost to produce a random mapping between elliptic curves in the same isomorphism class. This countermeasure has already been applied to elliptic curve cryptography as a defense against differential power analysis by Joye and Tymen in [15]. In this work, Joye and Tymen show that curves in the short Weierstrass form can easily be pushed through a random isomorphism by finding a random element u in the curve's underlying field and scaling both the curve coefficients and point

Algorithm 2 Proposed first round of SIDH protocol with a random pre-isogeny isomorphism to provide exposure resistance from Alice's point of view

Input: SIDH protocol over short Weierstrass curve $E_0/\mathbb{F}_q: \ y^2 = x^3 + ax + b$
Alice's torsion basis: P_A,Q_A, Bob's torsion basis: P_B,Q_B
Alice's private keys m_A, n_A
1. Compute secret kernel point $R = [m_A]P_A + [n_A]Q_A$
2. Randomly choose an element $u \in \mathbb{F}_q$
3. Form new points P'_B, Q'_B, R' with the relation $(x', y') = (u^2x, u^3y)$
4. Find new curve E'_0 with $a' = u^4a$, $b' = u^6b$
5. Perform isogeny over kernel $\phi'_A: \ E'_0 \rightarrow E'_0/\langle R' \rangle = E'_A$
6. Evaluate torsion points over isogeny $\phi'_A(P_B), \phi'_A(Q_B)$
7. Return public keys $\{E'_A, \phi'_A(P_B), \phi'_A(Q_B)\}$

coordinates by some power of u. When applying this countermeasure to SIDH, the primary difference is that any isomorphic curve and corresponding torsion points will do as the j-invariant is the final shared secret. In Algorithm 2 we show how the isomorphism will be applied in accordance with SIDH during the first round. The second round is performed in a similar matter, but the other party's torsion bases are not pushed through the isogeny. One minor difference with the Joye and Tymen proposal is that we define the powers of u with positive powers rather than negative powers so that an inverse is not required.

As Algorithm 2 shows, the random pre-isogeny isomorphism requires the generation of a random element in $\mathbb{F}_q = \mathbb{F}_{p^2}$ as well as several field multiplications. Specifically, since supersingular elliptic curves can be defined over $\mathbb{F}_{p^2}$, approximately $2\log_2 p$ random bits must be generated from a true random number generator. A deterministic number generator would not suffice as an attacker could then determine which bits are used for the random isomorphism. Within the quadratic prime field $\mathbb{F}_{p^2}$, let us denote the cost to generate a random element as r, the cost of a field comparison as δ, the cost of a field inversion as I, the cost of a field multiplication as M, and the cost of a field squaring as S. Thus, this isomorphism countermeasure for the first round of SIDH has a cost of $r + 9M + 3S$ and the second round has a cost of $r + 5M + 3S$. Compared to the cost of a round of SIDH, this constant cost is insignificant, as a large-degree isogeny at 128 quantum security level could take more than 15,000 field multiplications [20].

We note that we defined this over the short Weierstrass curve as any other elliptic curve can be converted to the short Weierstrass form. We further note that the isomorphism is computed pre-isogeny and post-double point multiplication. One could opt to perform the isomorphism before the double point multiplication, but this would surrender certain efficiency gains that target fast forms of initial curves. For instance, [6] proposes the choice of a supersingular curve in the base curve that allows base field computations and a simple Montgomery ladder to greatly speed up the secret kernel point generation. Thus, by defining

the isomorphism as pre-isogeny, we do not affect the double point multiplication computation that generates the secret kernel. Nevertheless, the isomorphism could be performed pre-double point multiplication as the initial curve isomorphism has been a strategy to provide some defense against differential power analysis [15].

4.3 Considerations for Kirkwood et al. Validation Model

The Kirkwood et al. [16] validation model essentially acts as a protection for static key users to ensure that the opposite party is acting honestly. For SIDH this is necessary for security as it is extremely difficult to validate the public key parameters exchanged over a public channel. As [6] proposes, one can validate public key parameters by ensuring that the torsion basis points each have the correct order and are independent with the Weil pairing. Further, one must ensure that the supersingular elliptic curve is of the correct cardinality, is supersingular, and is in the correct supersingular isogeny class. However, this validation only ensures that the public parameters *appear* valid. Indeed, if the parameters do not adhere to this validation, they should be rejected.

However, this does not protect against all attacks. As noted by Galbraith et al. in [10], there is a simple oracle attack on an SIDH system using static keys. A malicious third-party can send Alice public parameters $\{E_B, \phi_B(P_A), \phi_B(Q_A)\}$ that seem reasonable, but $\{R = \phi_B(P_A), S = \phi_B(Q_A)\}$ are sent in the form $\{R - [x\ell^{n-i-1}]S, [1 + \ell^{n-i-1}]S\}$. If Alice generates a shared key that matches the oracle's prediction, i.e., upon using the key, then the third-party knows that m_i of Alice's public key is '1'.

Countermeasures to the above oracle attack include using ephemeral keys and utilizing the Kirkwood et al. [16] validation model. However, an SIDH user might not always have access to a random number generator to generate new keys and this is also costly as a new key must be used for every key agreement. Let us assume that Alice is using a static key for SIDH. The Kirkwood et al. validation model ensures Bob is honestly producing the ephemeral keys he sends to Alice. In this model, Bob uses a pseudo-random function with seed r_B to generate his secret keys, generates the shared secret with Alice's public parameters, and sends Alice his seed encrypted with a key derivation function based on the shared secret. Alice finishes the protocol on her side with Bob's public information and uses the shared secret as an input to a key derivation function to decrypt Bob's seed. Alice then uses the retrieved private keys to verify that Bob performed the first round of SIDH honestly. If Alice's derived public parameters for Bob do not match the public parameters Bob sent, then Alice rejects the shared secret.

However, with the introduction of the random pre-isogeny curve isomorphism, Bob's large-degree isogeny computation would produce a random final curve within the correct isomorphism class. In this case, Alice will have to determine by some means whether Bob's resulting public keys are honestly generated. Bob could release the random curve isomorphism he used, but this would defeat the whole purpose of the isomorphism, as nothing is hidden in the exposure model. Alice's two options are:

1. Find some means to validate the public keys
2. Force both parties to perform the inverse of the pre-isogeny isomorphism at the end of the large-degree isogeny

Validating the public keys for Kirkwood et al. validation model. As for the first option, validating the public keys by some other means is still an interesting problem. The only upside is that in this case Alice has Bob's private key. Let us assume that $\{E_B, \phi_B(P_A), \phi_B(Q_A)\}$ is the golden set of public keys that would be generated by Bob if no pre-isogeny isomorphism is applied. Next, let us assume that Bob did use his own random pre-isogeny isomorphism and arrived at $\{E'_B, \phi'_B(P_A), \phi'_B(Q_A)\}$, which he sent to Alice over a public channel. With Bob's private key, Alice will perform the Bob's first round and also utilize a pre-isogeny isomorphism to obtain $\{E''_B, \phi''_B(P_A), \phi''_B(Q_A)\}$. At this point, Alice has two sets of public keys, $\{E'_B, \phi'_B(P_A), \phi'_B(Q_A)\}$ from Bob and $\{E''_B, \phi''_B(P_A), \phi''_B(Q_A)\}$ that she generated from Bob's supposed private key. Alice can easily verify that the curves E'_B are E''_B are in the same isomorphism class because they will share the same j-invariant, i.e., $j(E'_B) = j(E''_B)$. However, verifying the torsion basis points remains a problem. Similar to the key validation proposed in [6], one can check that both sets of torsion points have the correct order and are independent with the Weil pairing, but this does not protect against the oracle attack proposed in [10].

In order to determine if the basis points were honestly generated, Alice could perform an additional isomorphism from E'_B to E''_B and check if the torsion points match. Since Alice has been given a pair of curves E'_B and E''_B and a pair of points $\phi'_B(P_A)$ and $\phi''_B(P_A)$ with the claim that E'_B is isomorphic to E''_B and $\phi'_B(P_A)$ maps to $\phi''_B(P_A)$ under this isomorphism, we can verify this claim by finding the unique isomorphism between the two curves and verifying that the points do indeed map to each other. More specifically, let ψ be the isomorphism from E'_B to E''_B. Alice can apply this isomorphism to the torsion basis points $\phi'_B(P_A)$ and $\phi'_B(Q_A)$ and check that:

$$\psi(\phi'_B(P_A)) = \phi''_B(P_A),$$

$$\psi(\phi'_B(Q_A)) = \phi''_B(Q_A) \tag{1}$$

If these points match, then Alice indeed knows that Bob's public keys were honestly generated. If the torsion points do not properly match under the isomorphism, then Alice knows that Bob was not performing his half of SIDH honestly, and can reject Bob's session.

Both parties will perform an inverse isomorphism at the end of the large-degree isogeny. As for the second option, the protocol can call for both parties to provide the golden set of public keys. In this case, if Alice's golden set does not match Bob's, then she knows that Bob is acting dishonestly. Determining which set of Vélu's formulas are used to determine the golden set is a conversation between Alice and Bob. If Alice or Bob intend to perform a

pre-isogeny isomorphism, then they must perform a final inverse isomorphism to arrive back at the golden set.

Let us assume that Alice will use the random pre-isogeny isomorphism. In this case, Alice will still arrive at the correct isomorphism class. Since Alice's curve E''_B and the golden curve E_B are in the same isomorphism class, there exists a unique isomorphism that will produce the expected set of public keys. Notationwise, Alice performed the pre-isogeny isomorphism $\psi : E_0 \to E''_0$ followed by the isogeny $\phi_B : E''_0 \to E''_B$ and must now find some isomorphism $\psi^{-1} : E''_B \to E_B$. With this isomorphism, Alice can easily check the torsion points as:

$$\psi^{-1}(\phi''_B(\psi(P_A))) = \phi_B(P_A),$$

$$\psi^{-1}(\phi''_B(\psi(Q_A))) = \phi_B(Q_A) \tag{2}$$

Unfortunately, determining this unique isomorphism is not very simple as Alice does not know what the golden curve should be. One possibility is that Alice could compute ψ^{-1} in the initial isomorphism class and track ψ^{-1} in each new isomorphism class as the large-degree isogeny is performed. This is very costly as now some extra calculations must be performed at each isogeny, so this fix for the Kirkwood et al. validation model now scales with the complexity of the isogeny.

4.4 Countermeasure Costs to Comply with Kirkwood et al. Validation Model

Among Alice's two options above, directly performing the elliptic curve isomorphism between the two sets of public keys is by far the cheaper option. As was noted, the strategy to determine ψ^{-1} at each isomorphism class scales with the complexity of the isogeny rather than provide a constant cost. Thus, here we examine Alice's cost to perform that isomorphism and verify that Bob's public keys were produced honestly.

As was noted above, there exists a unique isomorphism between the public keys that Bob sent, $\{E'_B, \phi'_B(P_A), \phi'_B(Q_A)\}$, and Alice's computed public keys with Bob's seeded private keys, $\{E''_B, \phi''_B(P_A), \phi''_B(Q_A)\}$. In the simplest of ways, we again refer to the analysis by Joye and Tymen in [15] that was used to generate a random isomorphism. Here, Alice can solve for an element u that acts as the map between E'_B and E''_B. Over short Weierstrass curves, let us denote E'_B with curve coefficients a' and b' and E''_B with curve coefficients a'' and b''. Then supposing that Bob was acting honestly and the two sets of public keys are mapped to each other, the following sets of equations must hold for some unknown element u:

$$a' = u^4 a'' \tag{3}$$

$$b' = u^6 b'' \tag{4}$$

$$x_{\phi''_B(P_A)} = u^2 x_{\phi'_B(P_A)},\ x_{\phi''_B(Q_A)} = u^2 x_{\phi'_B(Q_A)} \tag{5}$$

$$y_{\phi''_B(P_A)} = u^3 y_{\phi'_B(P_A)},\ y_{\phi''_B(Q_A)} = u^3 y_{\phi'_B(Q_A)} \tag{6}$$

Therefore, Alice can solve for u by utilizing any two equations and dividing through. For instance, by dividing the first equations in formulas 5 and 6,

$$\frac{u^3 y_{\phi'_B(P_A)}}{u^2 x_{\phi'_B(P_A)}} = \frac{y_{\phi''_B(P_A)}}{x_{\phi''_B(P_A)}} \rightarrow u = \frac{x_{\phi'_B(P_A)} y_{\phi''_B(P_A)}}{y_{\phi'_B(P_A)} x_{\phi''_B(P_A)}} \tag{7}$$

Thus, the cost to compute the isomorphism to validate Bob's public keys is $I+3M$. After which, Alice must check that Equations 3-6 are valid, which prove that Alice's computed curve and points are indeed maps of Bob's. The first parts of Equations 5 and 6 were used to find u so they are already validated. The cost to generate u^4 and u^6 from u is $M+2S$, after which they are used to scale a'' and b'' and check that Equations 3-6 hold. Overall, the additional cost to utilize this isomorphism countermeasure in conjunction with the Kirkwood et al. validation model is $4\delta + I + 6M + 2S$.

However, in the grand scheme of things, it has been typical to use projective coordinates to greatly speed up elliptic curve cryptography arithmetic. For SIDH, Costello et al. [6] utilize multiple levels of projectivization and perform a 4-way inverse at the end, so that only a single inversion is required for an entire round. Therefore, if we incorporate the inversion cost necessary to comply with the Kirkwood et al. validation model in a simultaneous inversion trick, we are absorbing the inversion cost in exchange for several multiplications. The exact number of additional multiplications to perform the larger inversion is entirely dependent on how many values are being inverted. For instance going from a single inversion to a 2-way inversion has a cost change of I to $I + 3M$. Thus, although the full cost is $4\delta + I + 6M + 2S$, the cost of inversion here is not necessarily the cost of performing a large exponentiation as there will most likely already be an inversion performed.

In summary, the total cost of the random pre-isogeny isomorphism with this Kirkwood et al. validation model consideration is $r + 4\delta + I + 15M + 5S$, which is only experienced by a static-key user aiming to validate the opposite party's public keys.

5 Conclusion

In this work, we presented an exposure model for the supersingular isogeny Diffie-Hellman and proposed an additional protection against exposed values. By performing a random isomorphism just before the isogeny computation in isogeny-based cryptography, any intermediate elliptic curves or points that are divulged by any means are effectively obfuscated. We have shown that this countermeasure is relatively inexpensive and does not have any negative impacts on the protocol or validation of public keys with the Kirkwood et al. validation

model. Since isogeny-based cryptography is still in its infancy, there are other attacks that will most likely be discovered in the near-future, some of which may be implementation specific. The goal of this paper was to show that including this additional security precaution may be beneficial to long-term SIDH implementations.

6 Acknowledgement

The authors would like to thank the reviewers for their comments. Also, the authors would like to thank Dr. Luca De Feo for discussion and feedback. This work is supported in part by the grants NIST-60NANB17D184, NIST-60NANB16D246, ARO W911NF-17-1-0311, and NSF CNS-1661557, as well as CryptoWorks21, Public Works and Government Services Canada, Canada First Research Excellence Fund, and an RBC Fellowship.

References

1. Reza Azarderakhsh, Dieter Fishbein, and David Jao. Efficient Implementations of a Quantum-Resistant Key-Exchange Protocol on Embedded Systems. Technical report, University of Waterloo, 2014.
2. Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. Key Compression for Isogeny-Based Cryptosystems. In *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*, AsiaPKC '16, pages 1–10, ACM, 2016.
3. Reza Azarderakhsh, David Jao, and Christopher Leonardi. Post-Quantum Static-Static Key Agreement Using Multiple Protocol Instances. In *Selected Areas in Cryptography – SAC 2017, 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, pages 45–63, Springer International Publishing, 2018.
4. Lily Chen and Stephen Jordan. Report on Post-Quantum Cryptography. 2016. NIST IR 8105.
5. Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient Compression of SIDH Public Keys. In *Advances in Cryptology – EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 – May 4, 2017, Proceedings, Part I*, pages 679–706, Springer International Publishing, 2017.
6. Craig Costello, Patrick Longa, and Michael Naehrig. Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 572–601, 2016.
7. Luca De Feo, David Jao, and Jérôme Plût. Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, September 2014.
8. Harold M. Edwards. A Normal Form for Elliptic Curves. In *Bulletin of the American Mathematical Society*, 44, pages 393–422, 2007.

9. Steven Galbraith and Anton Stolbunov. Improved Algorithm for the Isogeny Problem for Ordinary Elliptic Curves. *Applicable Algebra in Engineering, Communication and Computing*, 24(2):107–131, June 2013.
10. Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the Security of Supersingular Isogeny Cryptosystems. In *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, pages 63–91, 2016.
11. Steven D. Galbraith, Christophe Petit, and Javier Silva. Signature Schemes Based On Supersingular Isogeny Problems. Cryptology ePrint Archive, Report 2016/1154, 2016.
12. Alexandre Gélin and Benjamin Wesolowski. Loop-Abort Faults on Supersingular Isogeny Cryptosystems. In *Post-Quantum Cryptography : 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings*, pages 93–106, Springer International Publishing, 2017.
13. Amir Jalali, Reza Azarderakhsh, and Mehran Mozaffari Kermani. Efficient Post-Quantum Undeniable Signature on 64-Bit ARM. In *Selected Areas in Cryptography – SAC 2017, 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, pages 281–298, Springer International Publishing, 2018.
14. David Jao and Luca De Feo. Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies. In *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 – December 2, 2011. Proceedings*, pages 19–34, Springer Berlin Heidelberg, 2011.
15. Marc Joye and Christophe Tymen. Protections Against Differential Analysis for Elliptic Curve Cryptography — An Algebraic Approach —. In *Cryptographic Hardware and Embedded Systems — CHES 2001: Third International Workshop Paris, France, May 14–16, 2001 Proceedings*, pages 377–390, Springer Berlin Heidelberg, 2001.
16. Daniel Kirkwood, Bradley C. Lackey, John McVey, Mark Motley, Jerome A. Solinas, and David Tuller. Failure is not an Option: Standardization Issues for Post-Quantum Key Agreement. Technical report, Workshop on Cybersecurity in a Post-Quantum World, 2015.
17. Brian Koziel, Reza Azarderakhsh, and David Jao. Side-Channel Attacks on Quantum-Resistant Supersingular Isogeny Diffie-Hellman. In *Selected Areas in Cryptography – SAC 2017, 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, pages 64–81, Springer International Publishing, 2018.
18. Brian Koziel, Reza Azarderakhsh, and Mehran Mozaffari-Kermani. Fast Hardware Architectures for Supersingular Isogeny Diffie-Hellman Key Exchange on FPGA. In *Progress in Cryptology – INDOCRYPT 2016: 17th International Conference on Cryptology in India, Kolkata, India, December 11-14, 2016, Proceedings*, pages 191–206, Springer International Publishing, 2016.
19. Brian Koziel, Reza Azarderakhsh, Mehran Mozaffari-Kermani, and David Jao. Post-Quantum Cryptography on FPGA Based on Isogenies on Elliptic Curves. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 64(1):86–99, January 2017.
20. Brian Koziel, Amir Jalali, Reza Azarderakhsh, David Jao, and Mehran Mozaffari-Kermani. NEON-SIDH: Efficient Implementation of Supersingular Isogeny Diffie-Hellman Key Exchange Protocol on ARM. In *Cryptology and Network Security:*

15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings, pages 88–103, Springer International Publishing, 2016.
21. Peter L. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Mathematics of Computation*, pages 243–264, 1987.
22. Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer, New York, 1992.
23. Seiichiro Tani. Claw Finding Algorithms Using Quantum Walk. *Theoretical Computer Science*, 410(50):5285–5297, November 2009.
24. Yan Bo Ti. Fault Attack on Supersingular Isogeny Cryptosystems. In *Post-Quantum Cryptography : 8th International Workshop, PQCrypto 2017, Utrecht, The Netherlands, June 26-28, 2017, Proceedings*, pages 107–122, Springer International Publishing, 2017.
25. Jacques Vélu. Isogénies Entre Courbes Elliptiques. *Comptes Rendus de l'Académie des Sciences Paris Séries A-B*, 273:A238–A241, 1971.
26. Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A Post-quantum Digital Signature Scheme Based on Supersingular Isogenies. In *Financial Cryptography and Data Security: 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*, pages 163–181, Springer International Publishing, 2017.